

Dynamisch toevoegen van nieuwe gegevens in een StUF-bericht

Inleiding

In deze notitie wordt een voorstel gedaan voor het dynamisch opnemen van extra gegevens in StUF-berichten. De nieuwe methode maakt het onder meer mogelijk om een willekeurige entiteit (bijvoorbeeld een Zaak) dynamisch te relateren aan andere entiteiten die mogelijk uit andere sectormodellen afkomstig zijn. Deze nieuwe aanpak vereist een paar kleine uitbreidingen op de StUF-onderlaag die backwards compatible zijn.

Alle schema's en xml-voorbeelden die in deze notitie worden gebruikt zijn ook te vinden in het bestand 'zkn0310_20130911.zip' op de StUF Community.

Uitbreidingen op de StUF 3.01 onderlaag

Het nieuwe element "extraElements"

In de huidige StUF 3.01 standaard, in het bijzonder het stuf0310.xsd schema, kennen we al het element "extraElementen" voor het dynamisch toevoegen van nieuwe gegevens zonder dat je het schema hoeft aan te passen:

```
<element name="extraElementen" type="StUF:ExtraElementen"/>
<complexType name="ExtraElementen">
  <sequence>
    <element name="extraElement" nillable="true" maxOccurs="unbounded">
      <complexType>
        <simpleContent>
          <extension base="string">
            <attributeGroup ref="StUF:element"/>
            <attribute name="naam" type="string" use="required"/>
            <attribute ref="StUF:indOnvolledigeDatum"/>
          </extension>
        </simpleContent>
      </complexType>
    </element>
  </sequence>
</complexType>
```

Het nadeel van deze constructie is dat je de nieuwe elementen niet kunt valideren tegen een schema en dat je geen hergebruik kunt maken van gegevensdefinities uit andere sectormodellen. Deze problemen kunnen we ondervangen door de introductie van een nieuw element met een subtiel andere naam:

```
<element name="extraElements" type="anyType"/>
```

Dit element is van het type "anyType" en kan daardoor willekeurige XML-expressies bevatten. Voor onze doeleinden is deze vrijheid iets te groot en eisen we dat de inhoud van dit element altijd moet voldoen aan de volgende structuur:

```
<stuf:extraElements>
  <ns1:extraElements stuf:schemaLocation="..." xmlns:ns1="...">...</ns1:extraElements>
</stuf:extraElements>
```

```

<ns2:extraElements stuf:schemaLocation="..." xmlns:ns2="...">...</ns2:extraElements>
...
<nsN:extraElements stuf:schemaLocation="..." xmlns:nsN="...">...</nsN:extraElements>
</stuf:extraElements>

```

Dit betekent dat op het hoogste niveau binnen `<stuf:extraElements>` alleen elementen mogen voorkomen met de naam “extraElements” maar dan wel uit andere namespaces. Het element `<stuf:extraElements>` valideert altijd ongeacht de inhoud. Dus het schema waarin dit element wordt gebruikt hoeft nooit te worden aangepast. Echter je kunt de subelementen `<ns1:extraElements>`, `<ns2:extraElements>`, ..., `<nsN:extraElements>` wel apart valideren tegen het schema dat gespecificeerd is in het attribute `stuf:schemaLocation`. Het enige wat dan gedaan hoeft te worden is het hernoemen van de prefix `stuf` naar `xsi` in het attribute `schemaLocation`. Bijvoorbeeld, het element

```

<ns1:extraElements stuf:schemaLocation="..." xmlns:ns1="...">...</ns1:extraElements>

```

moet als volgt worden aangepast

```

<ns1:extraElements xsi:schemaLocation="..." xmlns:ns1="...">...</ns1:extraElements>

```

voordat het aan de validator kan worden aangeboden. Deze omzetting is in een paar regels code te automatiseren. De truuk van het gebruik van de prefix `stuf` als dummy namespace van het attribute `schemaLocation` zorgt ervoor dat de inhoud van `<stuf:extraElements>` niet gevalideerd wordt in het hoofdschema. Dit is de basisconstructie om binnen een StUF-bericht dynamisch nieuwe gegevens op te nemen die volledig backward compatible is. Bovendien kunnen de nieuwe gegevens apart gevalideerd worden door ander schema's. Verderop in deze notitie zal deze nieuwe constructie worden toegelicht door middel van een concreet voorbeeld.

Het nieuwe attribute “functie”

Voor het eenvoudig linken naar andere sectormodellen hebben we nog een toevoeging nodig in het `stuf0310.xsd` schema. Het betreft de introductie van het attribute ‘`stuf:functie`’ waarmee we gerelateerde entiteiten direct kunnen linken zonder de overhead van een intermediaire relatie-entiteiten (zie http://www.kinggemeenten.nl/media/532655/stuf_functie_attribuut.pdf):

```

<attribute name="functie" type="StUF:FunctieElement"/>
<simpleType name="FunctieElement">
  <restriction base="string">
    <enumeration value="antwoord"/>
    <enumeration value="complex"/>
    <enumeration value="container"/>
    <enumeration value="gerelateerde"/>
    <enumeration value="groep"/>
    <enumeration value="entiteit"/>
    <enumeration value="onderdeel"/>
    <enumeration value="relatie"/>
    <enumeration value="selectie"/>
    <enumeration value="simpel"/>
    <enumeration value="update"/>
  </restriction>
</simpleType>

```

Backwards compatibility

De bovengenoemde uitbreidingen op de StUF 3.01 onderlaag zijn backwards compatible. In principe zouden we deze aanpassingen als een patch of iets dergelijks (dus zonder consequenties voor het versienummer en de namespace) kunnen doorvoeren. Het spreekt voor zich dat deze wijzigingsvoorstellen eerst beoordeeld moeten worden door de StUF Expertgroep.

Een voorbeeld met zaakspecifieke gegevens

Extra elementen voor het zaaktype “kapvergunning”

In de folder “ztc0100/eigenschappen” van ‘zkn0310_20130911.zip’ bevindt zich het schema “zkt0100_eig_kapvergunning”:

```
<element name="extraElements" type="kv:ExtraElements"/>
<complexType name="ExtraElements">
  <sequence>
    <element ref="kv:boom"/>
    <element ref="kv:eigenaar"/>
    <element name="geometrie" type="BG:GeometrieVlak-e" minOccurs="0"/>
  </sequence>
</complexType>
<element name="boom" type="GV:BOOM-basis"/>
<element name="eigenaar" type="BG:NPS-kerngegevens"/>
```

Dit schema specificeert de eigenschappen van het zaaktype “kapvergunning” door middel van het element “extraElements” en heeft een eigen (sub)namespace

```
xmlns:kv="http://www.egem.nl/StUF/sector/ztc/0100/eigenschappen/kapvergunning"
```

binnen de namespace van het sectormodel StUF-ZTC 1.0:

```
xmlns:ztc="http://www.egem.nl/StUF/sector/ztc/0100".
```

In de extra elementen wordt gebruik gemaakt van definities uit andere sectormodellen zoals StUF-GV 1.0 (Groen Voorziening) en StUF-BG 3.10 (Basis Gegevens), respectievelijk te zien aan de namespaces GV en BG in de prefixes van de types.

Extra elementen voor het entiteitstype ZAK

In de folder “zkn0310/entiteiten/extraElements” van het sectormodel StUF-ZKN 3.10 bevindt zich het schema “zkn0310_zak_extraElements.xsd”:

```
<element name="extraElements" type="zak:ExtraElements"/>
<complexType name="ExtraElements">
  <sequence>
    <element name="confidentieel" type="boolean"/>
    <element name="complexiteit" type="zak:Complexiteit"/>
  </sequence>
</complexType>
<simpleType name="Complexiteit">
  <restriction base="string">
    <enumeration value="makkelijk"/>
    <enumeration value="middelmatic"/>
    <enumeration value="moeilijk"/>
  </restriction>
</simpleType>
```

Dit schema bevat twee extra elementen voor het entiteitstype ZAK. Dit verschilt met de vorige sectie in de zin dat deze specifieke gegevens kunnen worden opgenomen bij alle berichten met het entiteitstype ZAK. De extra elementen die ter sprake kwamen in de vorige sectie kunnen alleen gebruikt worden bij berichten met het zaaktype “kapvergunning”.

Een voorbeeld

In de folder ‘zkn0310/voorbeeldbericht’ van het sectormodel StUF-ZKN 3.10 bevindt zich het bestand ‘zkn0310_zkt_kapvergunning.xml’:

```

<zkn:object stuf:entiteittype="ZAK" stuf:functie="entiteit"
xsi:schemaLocation="http://www.egem.nl/StUF/sector/zkn/0310 ../entiteiten/zkn0310_ent_basis.xsd"
xmlns:zkn="http://www.egem.nl/StUF/sector/zkn/0310" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:stuf="http://www.egem.nl/StUF/StUF0301" xmlns:bg="http://www.egem.nl/StUF/sector/bg/0310"
xmlns:gv="http://www.egem.nl/StUF/sector/gv0100">
  <zkn:identificatie>12345</zkn:identificatie>
  <zkn:omschrijving>Aanvraag kapvergunning</zkn:omschrijving>
  <zkn:isVan>
    <zkn:gerelateerde stuf:entiteittype="ZKT">
      <zkn:omschrijving>Kapvergunning</zkn:omschrijving>
      <zkn:code>100</zkn:code>
    </zkn:gerelateerde>
  </zkn:isVan>
  <stuf:extraElements>
    <kv:extraElements
stuf:schemaLocation="http://www.egem.nl/StUF/sector/zkn/0310/zaakspecifiek/kapvergunning
zkn0310_zkt_kapvergunning.xsd" xmlns:stuf="http://www.egem.nl/StUF/StUF0301"
xmlns:kv="http://www.egem.nl/StUF/sector/zkn/0310/zaakspecifiek/kapvergunning"
xmlns:bg="http://www.egem.nl/StUF/sector/bg/0310" xmlns:gv="http://www.egem.nl/StUF/sector/gv0100"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <kv:boom stuf:entiteittype="BOOM" stuf:functie="gerelateerde">
        <gv:soort>Eik</gv:soort>
        <gv:dikte>
          <gv:diameter>100</gv:diameter>
          <gv:eenheid>cm</gv:eenheid>
        </gv:dikte>
        <gv:lengte>4</gv:lengte>
      </kv:boom>
      <kv:eigenaar stuf:entiteittype="NPS" stuf:functie="gerelateerde">
        <bg:inp.bsn>123456789</bg:inp.bsn>
      </kv:eigenaar>
      <kv:confidentieel>true</kv:confidentieel>
    </kv:extraElements>
    <zak:extraElements
stuf:schemaLocation="http://www.egem.nl/StUF/sector/zkn/0310/extraElements/zak
../entiteit/extraElements/zkn0310_zak_extraElements.xsd"
xmlns:zak="http://www.egem.nl/StUF/sector/zkn/0310/extraElements/zak"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <zak:confidentieel>true</zak:confidentieel>
      <zak:complexiteit>moeilijk</zak:complexiteit>
    </zak:extraElements>
  </stuf:extraElements>
</zkn:object>

```

In dit object van entiteittype ZAK zijn zowel extra elementen opgenomen voor de eigenschappen van het zaaktype “kapvergunning” als de extra elementen van het entiteittype ZAK zelf. Het eerste geval correspondeert met het element `<kv:extraElements ...>` en het tweede geval met `<zak:extraElements ...>`. Beide elementen kunnen apart gevalideerd worden door de schema’s die gespecificeerd zijn in het attribute `stuf:schemaLocation`. Deze schema’s zijn besproken in de vorige secties. Het enige wat gedaan hoeft te worden is de conversie van `stuf:schemaLocation` naar `xsi:schemaLocation`. Deze twee losse validatiestappen kunnen worden uitgevoerd met de bestanden ‘zkn0310_zkt_kapvergunning_kv.xml’ en ‘zkn0310_zkt_kapvergunning_zak.xml’ in de folder ‘zkn0310/voorbeeldbericht’.